	Application No.	**Applicant(s)**
Examiner-Initiated Interview Summary	10/081,000	FOMENKO, ANATOLI
	Examiner	**Art Unit**	
	Etienne P. LeRoux	2161	

All Participants:

Status of Application: *Allowed*

(1) *Etienne P. LeRoux*.

(3) _____.

(2) *Forrest Gunnison*.

(4) _____.

Date of Interview: *8 January 2007*

Time: *1:15 pm*

Type of Interview:
☒ Telephonic
☐ Video Conference
☐ Personal (Copy given to: ☐ Applicant   ☐ Applicant's representative)

Exhibit Shown or Demonstrated: ☐ Yes   ☐ No
If Yes, provide a brief description: .

Part I.

Rejection(s) discussed: ‧

Claims discussed:

Prior art documents discussed:

Part II.

SUBSTANCE OF INTERVIEW DESCRIBING THE GENERAL NATURE OF WHAT WAS DISCUSSED:
a telephone call was made to Applicant to propose amending claim 1 in order to advance prosecution. Applicant agreed to amend claim 1 by incorporation of claim 5.

Part III.

☐ It is not necessary for applicant to provide a separate record of the substance of the interview, since the interview directly resulted in the allowance of the application. The examiner will provide a written summary of the substance of the interview in the Notice of Allowability.

☐ It is not necessary for applicant to provide a separate record of the substance of the interview, since the interview did not result in resolution of all issues. A brief summary by the examiner appears in Part II above.

_____
(Examiner/SPE Signature)

_____
(Applicant/Applicant's Representative Signature – if appropriate)

10/081,000

Garden West Office Plaza, Suite 220
1900 Garden Road
Monterey, CA 93940
(831) 655-0880
(831) 655-0888 (Fax)

Gunnison, McKay & Hodgson, L.L.P.

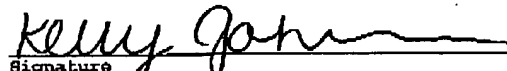|			
To:	Examiner Etienne P. LeRoux	From:	Forrest Gunnison
Fax:	1-571-273-4022	Pages:	11 Total
Phone:	1-571-272-4022	Date:	January 8, 2007
Re:	Response to Telephonic Request		
Your Ref:	Serial No. 10/081,000	Our Ref:	P-6507

Applicants: Anatoli Fomenko et al.
Assignee: Sun Microsystems, Inc.
Title: VERSIONING APPLICATION PROGRAMMING INTERFACE AND METHOD
              FOR USING VERSIONING FUNCTIONALITY
Serial No.:   10/081,000   Filed: February 20,
                                                 2002
Examiner: Etienne P. LeRoux   Group Art   2161
                                   Unit:
Docket No.:   P-6507

Enclosed is:

1) Response to Telephonic Request (10 Pages)

CERTIFICATION OF FACSIMILE TRANSMISSION
I hereby certify this paper is being facsimile transmitted to the Patent and
Trademark Office (fax No. 571-273-4022) on the date shown below:

Kelly John   January 8, 2007
Signature
Kelly Johnson

Appl. No. 10/081,000
Amdt. dated January 8, 2007
Reply to Telephonic Request of January 8, 2007

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

Applicant(s): Anatoli Fomenko

Assignee: Sun Microsystems, Inc.

Title: VERSIONING APPLICATION PROGRAMMING INTERFACE AND METHOD FOR USING VERSIONING FUNCTIONALITY

Serial No.: 10/081,000   Filed: February 20, 2002

Examiner: Etienne P. LeRoux   Group Art Unit: 2161

Docket No.: P-6507

Monterey, CA
January 8, 2007

RESPONSE TO TELEPHONIC REQUEST

Dear Sir:

Based on a telephonic conversation initiated by the Examiner on January 8, 2007, Applicant has moved the limitation of Claim 5 into Claim 1 and cancelled Claim 5 to place the application in condition for allowance. Claim 6 was amended to correct an informality created by the cancellation of Claim 5. A copy of the amended claims is attached.

Claims 1 to 4 and 6 to 32 remain in the application. Claims 1 and 6 have been amended. Claim 5 was canceled. For the foregoing reasons, Applicant(s) respectfully request allowance of all pending claims. If the Examiner has any questions relating to the above, the Examiner is respectfully requested to telephone the undersigned Attorney for Applicant(s).

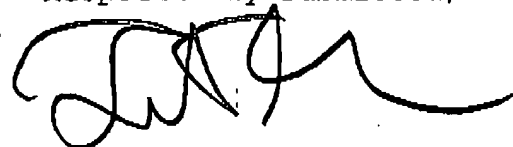**CERTIFICATE OF TRANSMISSION**
I hereby certify that this correspondence is being facsimile transmitted to the U.S. Patent and Trademark Office, Fax No. (571) 273-4022, on January 8, 2007.

Kelly Qah ———— January 8, 2007
Kelly Johnson   Date of Signature

Respectfully submitted,

Forrest Gunnison
Attorney for Applicant(s)
Reg. No. 32,899

Page 1 of 10

Appl. No. 10/081,000
Amdt. dated January 8, 2007
Reply to Telephonic Request of January 8, 2007

        This listing of claims replaces all prior versions, and
listings of claims in the instant application:

Listing of Claims:

        1.   (Currently Amended)   A versioning Application
Programming Interface (API) for a software platform based on an
object-oriented platform-independent programming language, said
versioning API comprising:
                main interfaces defining versioning functionality,
        said main interfaces allowing access to the versioning
        functionality;
                a functional implementation of said main interfaces,
        said functional implementation comprising classes and
        libraries implementing the versioning functionality, said
        classes including a reference to a program module to
        perform a requested versioning function;
                native programming interfaces allowing code written
        in the object-oriented platform-independent language to
        operate with code written in a native language other than
        the object-oriented platform-independent language; and
                a user interface for using the versioning
        functionality.

        2.   (Previously Presented)   The versioning API according
to claim 1, further comprising a communication mechanism
implementing client-server functionality.

        3.   (Previously Presented)   The versioning API according
to claim 1 wherein said main interfaces comprising:
                an interface-defining versioning server
        functionality;
                an interface defining versioning client
        functionality;

                            Page 2 of 10

an interface defining versioning repository
functionality;
an interface defining designated directory structures
and access to the designated directory structures; and
an interface defining transactions between the
designated directory structures.

4.   (Previously Presented)   The versioning API according
to claim 3 wherein said main interfaces further comprising:
an interface defining file actions within a
designated directory structure.

5.   (Cancelled)

6.   (Currently Amended)   The versioning API according to
claim 51 wherein said functional implementation comprising:
classes and first libraries written in an object-
oriented platform-independent programming language; and
second libraries including software routines written
in a native programming language other than the object-
oriented platform-independent language, said second
libraries implementing said native programming interfaces.

7.   (Previously Presented)   The versioning API according
to claim 6 wherein said classes comprise:
an implementation class including:
a reference to a first library, said reference being
invoked if a requested versioning function is implemented
with the object-oriented platform-independent programming
language; and
a reference to a native function and a second
library, said reference being invoked, using a native
programming interface, if a requested versioning function
is implemented with the native programming language.

CUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1900 Garden Road, Suite 220
Monterey, CA 93940
(831) 655-0880
Fax (831) 655-0888

Page 3 of 10

8.   (Previously Presented)   The versioning API according to claim 6 wherein said functional implementation further comprising:

resource files available to said classes and libraries.

9.   (Previously Presented)   The versioning API according to claim 1 wherein the classes comprise:

a class BringoverFrom including a reference to a program module for copying master files stored in a first directory structure and thereby creating a set of working files; and

a class BringoverTo including a reference to a program module for storing the set of working files in a second directory structure.

10.   (Previously Presented)   The versioning API according to claim 9 wherein the classes further comprise:

a class PutbackFrom including a reference to a program module for copying the working files in the second directory structure and thereby creating a set of updated files; and

a class PutbackTo including a reference to a program module for replacing a corresponding set of the master files in the first directory structure with the set of updated files.

11.   (Previously Presented)   The versioning API according to claim 10 wherein the classes further comprise:

a class Conflict including:

a reference to a program module for receiving a request for replacing the master files with a set of updated files, and checking for a previous replacement of the master files with another set of updated files;

Page 4 of 10

Appl. No. 10/081,000
Amdt. dated January 8, 2007
Reply to Telephonic Request of January 8, 2007

a reference to said class PutbackTo;

a reference to said class BringoverFrom; and a
reference to said class BringoverTo.

12. (Previously Presented) The versioning API according
to claim 11 wherein said reference to said class PutbackTo is
invoked if there is no previous replacement of the master
files.

13. (Previously Presented) The versioning API according
to claim 11 wherein said reference to said class BringoverFrom
and said reference to said class BringoverTo are invoked if
there is a previous replacement of the master files.

14. (Previously Presented) The versioning API according
to claim 10 wherein the classes further comprise:

·a class Checkout including:

a reference to a program module for creating a
writeable copy of a working file stored in the second
directory structure; and

a reference to a program module for storing the
writeable copy to a requested address; and
a class Checkin including:

a reference to a program module for copying the
writeable copy of a requested address so as to create
an updated working file; and

a reference to a program module for replacing
the working file with the updated working file.

15. (Previously Presented) The versioning API according
to claim 14 wherein the classes further comprise:

a class Lock including a reference to a program
module for receive a request for creating a writeable copy
of a working file and checking whether a writeable copy of
the working file has already been created.

Page 5 of 10

16.   (Previously Presented)   The versioning API according
to claim 9 wherein the classes further comprise:
        a class Freezepoint including a reference to a
program module for creating freezepoint files for files in
a specified directory structure, the freezepoint files
storing a specific time stamp and a then current version
of the corresponding files.

17.   (Previously Presented)   The versioning API according
to claim 1 wherein said user interface comprises at least one
of:
        a graphic user interface; and
        a command line interface.

18.   (Original)   A method for using version control
functionality via a versioning Application Programming
Interface (API) provided in a software platform based on an
object oriented platform-independent programming language, said
method comprising:
        defining versioning functionality in main interfaces
of said versioning API;
        implementing the versioning functionality in classes
and libraries of said versioning API, the libraries
including:
                first libraries written in an object-oriented
        platform-independent programming language, and
                second libraries written in a native programming
        language other than the object-oriented platform-
        independent language; and
                providing native programming interfaces allowing code
written in the object-oriented platform-independent
language to operate with code written in a native language
other than the object-oriented platform-independent

GUNNISON, McKAY &
HODGSON, L.L.P.
Garden West Office Plaza
1000 Garden Road, Suite 229
Monterey, CA 93940
(831) 655-0100
Fax (831) 655-0888

Page 6 of 10

language, the second libraries including native
programming interface implementation.

19.   (Previously Presented) The method according to claim
18, further comprising:

receiving, from a client, a request for a versioning
function;

calling a class implementing the requested versioning
function;

invoking a first library from the class, if the
requested versioning function is implemented in the first
library written in the object-oriented platform-
independent program language; and

using a native programming interface from the class,
so as to invoke a second library if a requested versioning
function is implemented in the second library written in a
native language other than the object-oriented platform-
independent language.

20.   (Previously Presented) The method according to claim
18 wherein the classes and libraries are mounted with a
versioning server application deployed to a hosting server
running on the software platform, the software platform being
based on the object-oriented platform-independent programming
language, said method further comprising:

making a call for a method of a proxy object at the
client, the proxy object being associated with a type of
versioning transaction;

converting the call for a method to a request of the
method;

transmitting the request to the hosting server; and

invoking a servlet at the hosting server to generate
a response to the request, the servlet delegating
processing of the request to a server object calling a
class including the requested method;

Page 7 of 10

invoking, from the class, the method directly if the requested method is implemented in a first library written in the object-oriented platform-independent program language; and

invoking, from the class, the method using a native programming interface if the requested method is implemented in a second library written in a native language other than the object-oriented platform-independent program language.

21. (Previously Presented) The method according to claim 20 wherein said making a call is performed with a graphic user interface.

22. (Previously Presented) The method according to claim 20 wherein said making a call is performed with a command line interface.

23. (Previously Presented) The method according to claim 18 wherein said defining versioning functionality comprising:
defining versioning server functionality;
defining versioning client functionality;
defining versioning repository functionality;
defining designated directory structures access to the designated directory structures; and
defining transactions between the designated directory structures.

24. (Previously Presented) The method according to claim 23 wherein said defining versioning functionality further comprising:
defining file actions within a designated directory structure.

Page 8 of 10

Appl. No. 10/081,000
Amdt. dated January 8, 2007
Reply to Telephonic Request of January 8, 2007

25. (Previously Presented) The method according to claim 18 wherein in said implementing, the versioning functionality is further implemented in resource files available to the classes and libraries.

26. (Previously Presented) The method according to claim 18 wherein the versioning functionality implemented in classes and libraries comprises:

copying master files stored in a first directory structure and thereby creating a set of working files; and
storing the set of working files in a second directory structure.

27. (Previously Presented) The method according to claim 26 wherein the versioning functionality implemented in classes and libraries further comprises:

copying the working files in the second directory structure and thereby creating a set of updated files; and
replacing the master files in the first directory structure with the set of updated files.

28. (Previously Presented) The method according to claim 27 wherein the versioning functionality implemented in classes and libraries further comprises:

receiving a request for replacing the master files with a set of updated files, and checking for a previous replacement of the master files with another set of updated files;
calling said replacing if there is no previous replacement of the master files since a previous copying of the master files; and
calling said copying master files and said storing if there is a previous replacement of the master files since a previous copying of the master files.

Page 9 of 10

Appl. No. 10/081,000
Amdt. dated January 8, 2007
Reply to Telephonic Request of January 8, 2007

29. (Previously Presented) The method according to claim 27 wherein the versioning functionality implemented in classes and libraries further comprises:

creating a writeable copy of a working file stored in the second directory structure; and

storing the writeable copy to a requested address.

30. (Previously Presented) The method according to claim 29 wherein the versioning functionality implemented in classes and libraries further comprises:

copying the writeable copy so as to create an updated working file; and

replacing the working file in the second directory with the updated working file.

31. (Previously Presented) The method according to claim 30 wherein the versioning functionality implemented in classes and libraries further comprises:

receive a request for creating a writeable copy of a working file; and

checking whether a writeable copy of the working file has already been created.

32. (Previously Presented) The method according to claim 27 wherein the versioning functionality implemented in classes and libraries further comprises:

creating freezepoint files for files in a specified directory structure, the freezepoint files storing a specific time stamp and a then current version of the corresponding files.

Page 10 of 10